Data Analysis of Talent Migration Patterns

Included dataset shows the movement of LinkedIn members across the world between 2015 and 2019.

Data source - World Bank https://datacatalog.worldbank.org/dataset/talent-migration-linkedin-data (https://datacatalog.worldbank.org/dataset/talent-migration-linkedin-data)

The **Net Migration** here refers to the net gain or loss of members from another country divided by the average LinkedIn membership of the target (or selected) country during the time period, multiplied by 10,000.

The analysis includes the following sections:

1. Data Understanding & Preparation
2. Data Visualisation
3. Creating Geographical Map
4. Extracting Insights from Armenian Talent Migration
5. Exploring Flow Directions (based on income groups)

In [1]:
```
import sys
sys.path.append('/usr/local/lib/python3.9/site-packages')
```

1. Data Understanding & Preparation

In [2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

In [3]:
```
import geopandas as gpd
import json
from bokeh.io import output_notebook, show, output_file
from bokeh.plotting import figure
from bokeh.models import GeoJSONDataSource, LinearColorMapper, ColorBar
from bokeh.palettes import brewer
```

In [4]:
```
data = pd.read_csv("country_migration_public.csv")
```

In [5]:
```
data.head()
```

Out[5]:

	base_country_code	base_country_name	base_lat	base_long	base_country_wb_income	base_country_wb_region	target_country_code	tar
0	ae	United Arab Emirates	23.424076	53.847818	High Income	Middle East & North Africa	af	
1	ae	United Arab Emirates	23.424076	53.847818	High Income	Middle East & North Africa	dz	
2	ae	United Arab Emirates	23.424076	53.847818	High Income	Middle East & North Africa	ao	
3	ae	United Arab Emirates	23.424076	53.847818	High Income	Middle East & North Africa	ar	
4	ae	United Arab Emirates	23.424076	53.847818	High Income	Middle East & North Africa	am	

5 rows × 26 columns

In [6]:
```
data.tail()
```

Out[6]:

	base_country_code	base_country_name	base_lat	base_long	base_country_wb_income	base_country_wb_region	target_country_code
4143	zw	Zimbabwe	-19.015438	29.154857	Low Income	Sub-Saharan Africa	za
4144	zw	Zimbabwe	-19.015438	29.154857	Low Income	Sub-Saharan Africa	ae
4145	zw	Zimbabwe	-19.015438	29.154857	Low Income	Sub-Saharan Africa	gb
4146	zw	Zimbabwe	-19.015438	29.154857	Low Income	Sub-Saharan Africa	us
4147	zw	Zimbabwe	-19.015438	29.154857	Low Income	Sub-Saharan Africa	zm

5 rows × 26 columns

In [7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4148 entries, 0 to 4147
Data columns (total 26 columns):
 #   Column                     Non-Null Count  Dtype  
---  ------                     --------------  -----  
 0   base_country_code          4148 non-null   object 
 1   base_country_name          4148 non-null   object 
 2   base_lat                   4148 non-null   float64
 3   base_long                  4148 non-null   float64
 4   base_country_wb_income     4148 non-null   object 
 5   base_country_wb_region     4148 non-null   object 
 6   target_country_code        4148 non-null   object 
 7   target_country_name        4148 non-null   object 
 8   target_lat                 4148 non-null   float64
 9   target_long                4148 non-null   float64
 10  target_country_wb_income   4148 non-null   object 
 11  target_country_wb_region   4148 non-null   object 
 12  net_per_10K_2015           4148 non-null   float64
 13  net_per_10K_2016           4148 non-null   float64
 14  net_per_10K_2017           4148 non-null   float64
 15  net_per_10K_2018           4148 non-null   float64
 16  net_per_10K_2019           4148 non-null   float64
 17  Unnamed: 17                0 non-null      float64
 18  Unnamed: 18                0 non-null      float64
 19  Unnamed: 19                0 non-null      float64
 20  Unnamed: 20                0 non-null      float64
 21  Unnamed: 21                0 non-null      float64
 22  Unnamed: 22                0 non-null      float64
 23  Unnamed: 23                0 non-null      float64
 24  Unnamed: 24                0 non-null      float64
 25  Unnamed: 25                0 non-null      float64
dtypes: float64(18), object(8)
memory usage: 842.7+ KB
```

In [8]:
```
data.columns
```

Out[8]:
```
Index(['base_country_code', 'base_country_name', 'base_lat', 'base_long',
       'base_country_wb_income', 'base_country_wb_region',
       'target_country_code', 'target_country_name', 'target_lat',
       'target_long', 'target_country_wb_income', 'target_country_wb_region',
       'net_per_10K_2015', 'net_per_10K_2016', 'net_per_10K_2017',
       'net_per_10K_2018', 'net_per_10K_2019', 'Unnamed: 17', 'Unnamed: 18',
       'Unnamed: 19', 'Unnamed: 20', 'Unnamed: 21', 'Unnamed: 22',
       'Unnamed: 23', 'Unnamed: 24', 'Unnamed: 25'],
      dtype='object')
```

Renaming some columns

In [9]:
```
cols= ['net_per_10K_2015','net_per_10K_2016', 'net_per_10K_2017','net_per_10K_2018', 'net_per_10K_2019']
```

In [10]:
```
cols_ = {}
for i in cols:
    cols_[i] = i.split('_')[-1]
```

In [12]:
```
data.rename(columns=cols_, inplace=True)
```

Getting rid of empty columns

In [14]:
```
cols_to_drop = ['base_lat', 'base_long','target_lat',
       'target_long','Unnamed: 17', 'Unnamed: 18',
       'Unnamed: 19', 'Unnamed: 20', 'Unnamed: 21', 'Unnamed: 22',
       'Unnamed: 23', 'Unnamed: 24', 'Unnamed: 25']
for col in cols_to_drop:
    data.drop(col, axis=1, inplace=True)
```

In [15]:
```
countries = data['base_country_name'].unique()
```

In [16]:
```
f"The dataset includes {len(countries)-1} countries."
```

Out[16]: 'The dataset includes 139 countries.'

In [18]:
```
data
```

Out[18]:

	base_country_code	base_country_name	base_country_wb_income	base_country_wb_region	target_country_code	target_country_name
0	ae	United Arab Emirates	High Income	Middle East & North Africa	af	Afghanistan
1	ae	United Arab Emirates	High Income	Middle East & North Africa	dz	Algeria
2	ae	United Arab Emirates	High Income	Middle East & North Africa	ao	Angola
3	ae	United Arab Emirates	High Income	Middle East & North Africa	ar	Argentina
4	ae	United Arab Emirates	High Income	Middle East & North Africa	am	Armenia
...	...	...	...	...	...	...
4143	zw	Zimbabwe	Low Income	Sub-Saharan Africa	za	South Africa
4144	zw	Zimbabwe	Low Income	Sub-Saharan Africa	ae	United Arab Emirates
4145	zw	Zimbabwe	Low Income	Sub-Saharan Africa	gb	United Kingdom
4146	zw	Zimbabwe	Low Income	Sub-Saharan Africa	us	United States
4147	zw	Zimbabwe	Low Income	Sub-Saharan Africa	zm	Zambia

4148 rows × 13 columns

In [17]:
```
data.describe()
```

Out[17]:

	2015	2016	2017	2018	2019
count	4148.000000	4148.000000	4148.000000	4148.000000	4148.000000
mean	0.461757	0.150248	-0.080272	-0.040591	-0.022743
std	5.006530	4.201118	3.203092	3.593876	3.633247
min	-37.010000	-40.890000	-43.660000	-56.220000	-50.330000
25%	-0.150000	-0.190000	-0.210000	-0.210000	-0.210000
50%	0.000000	0.000000	0.000000	0.000000	0.000000
75%	0.240000	0.220000	0.160000	0.170000	0.180000
max	150.680000	124.480000	87.000000	91.410000	87.710000

In [19]:
```
data.loc[data['2015'] == 150.680000]
```

Out[19]:

	base_country_code	base_country_name	base_country_wb_income	base_country_wb_region	target_country_code	target_country_name
2487	lu	Luxembourg	High Income	Europe & Central Asia	fr	France

The biggest inflow (2015-2019) was in Luxembourd from France in 2015.

In [20]:
```
data.loc[data['2018'] == -56.220000]
```

Out[20]:

	base_country_code	base_country_name	base_country_wb_income	base_country_wb_region	target_country_code	target_country_name
3658	tn	Tunisia	Lower Middle Income	Middle East & North Africa	fr	France

The biggest ouflow (2015-2019) was from Tunisia to France in 2018.

In [21]:
```
data.dtypes
```

Out[21]:
```
base_country_code            object
base_country_name            object
base_country_wb_income       object
base_country_wb_region       object
target_country_code          object
target_country_name          object
target_country_wb_income     object
target_country_wb_region     object
2015                        float64
2016                        float64
2017                        float64
2018                        float64
2019                        float64
dtype: object
```

In [22]:
```
data.isna().sum()
```

Out[22]:
```
base_country_code           0
base_country_name           0
base_country_wb_income      0
base_country_wb_region      0
target_country_code         0
target_country_name         0
target_country_wb_income    0
target_country_wb_region    0
2015                        0
2016                        0
2017                        0
2018                        0
2019                        0
dtype: int64
```

In [23]:
```
data.columns
```

Out[23]:
```
Index(['base_country_code', 'base_country_name', 'base_country_wb_income',
       'base_country_wb_region', 'target_country_code', 'target_country_name',
       'target_country_wb_income', 'target_country_wb_region', '2015', '2016',
       '2017', '2018', '2019'],
      dtype='object')
```

Grouping and creating new dataframes for each country and region

In [25]:
```
dt = data.groupby('base_country_name').sum().reset_index()
```

In [26]:
```
dt
```

Out[26]:

	base_country_name	2015	2016	2017	2018	2019
0	Afghanistan	6.54	5.24	-34.27	15.85	23.01
1	Albania	4.78	0.59	-18.80	-5.05	-16.58
2	Algeria	-5.31	-12.26	-34.10	-23.00	-23.47
3	Angola	73.98	13.90	-9.24	4.62	19.07
4	Argentina	-0.20	2.14	8.19	14.72	-8.77
...	...	...	...	...	...	...
135	Vietnam	4.75	-6.52	-15.12	-9.08	-7.94
136	West Bank and Gaza	6.75	-5.46	-11.79	-14.13	-15.59
137	Yemen, Rep.	-10.27	-11.99	-7.16	-4.38	2.89
138	Zambia	93.77	64.92	20.12	23.66	27.05
139	Zimbabwe	45.83	38.31	0.97	-9.03	-23.98

140 rows × 6 columns

In [27]:
```
region_dt = data.groupby('base_country_wb_region').sum().reset_index()
```

In [28]:
```
region_dt
```

Out[28]:

	base_country_wb_region	2015	2016	2017	2018	2019
0	East Asia & Pacific	341.14	189.28	94.83	133.17	132.22
1	Europe & Central Asia	160.59	64.38	73.43	226.29	354.42
2	Latin America & Caribbean	122.95	-109.09	-250.61	-371.23	-375.41
3	Middle East & North Africa	437.01	69.57	-108.09	-9.48	-59.21
4	North America	11.86	26.79	37.98	50.69	60.29
5	South Asia	-66.53	-99.02	-139.47	-101.33	-103.43
6	Sub-Saharan Africa	908.35	481.32	-41.04	-96.48	-103.22

In [29]:
```
regions_transposed = region_dt.set_index('base_country_wb_region').transpose()
```

In [30]:
```
regions_transposed
```

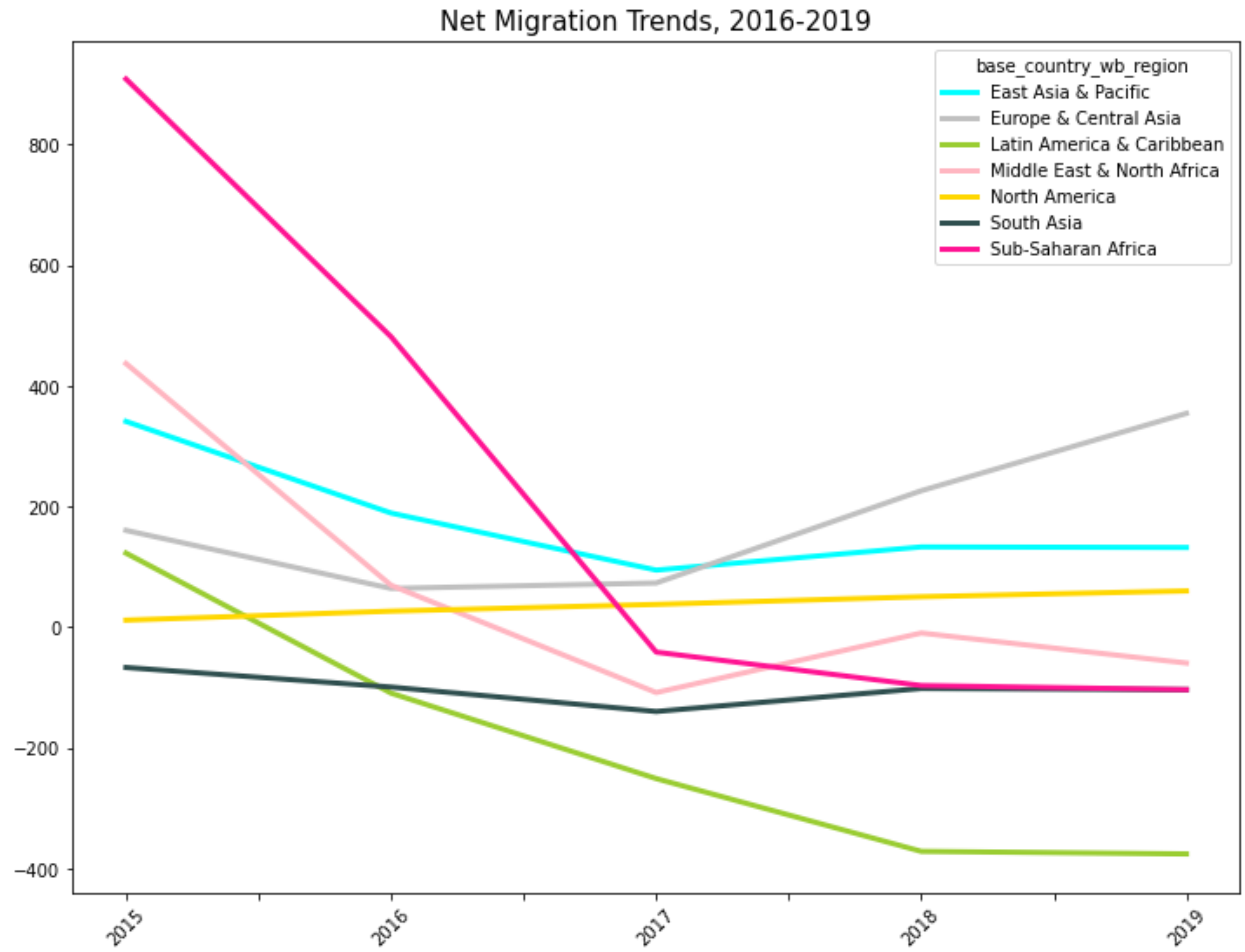
Out[30]:

base_country_wb_region	East Asia & Pacific	Europe & Central Asia	Latin America & Caribbean	Middle East & North Africa	North America	South Asia	Sub-Saharan Africa
2015	341.14	160.59	122.95	437.01	11.86	-66.53	908.35
2016	189.28	64.38	-109.09	69.57	26.79	-99.02	481.32
2017	94.83	73.43	-250.61	-108.09	37.98	-139.47	-41.04
2018	133.17	226.29	-371.23	-9.48	50.69	-101.33	-96.48
2019	132.22	354.42	-375.41	-59.21	60.29	-103.43	-103.22

2. Data Visualisation

In [31]:
```
cls = ['cyan', 'silver','yellowgreen','lightpink', 'gold', 'darkslategrey','deeppink','palevioletred']
regions_transposed.plot.line(figsize=(12,9), color =cls, lw = 3)

plt.xticks(rotation=45)
plt.title('Net Migration Trends, 2016-2019', fontsize=15)
plt.show()
```



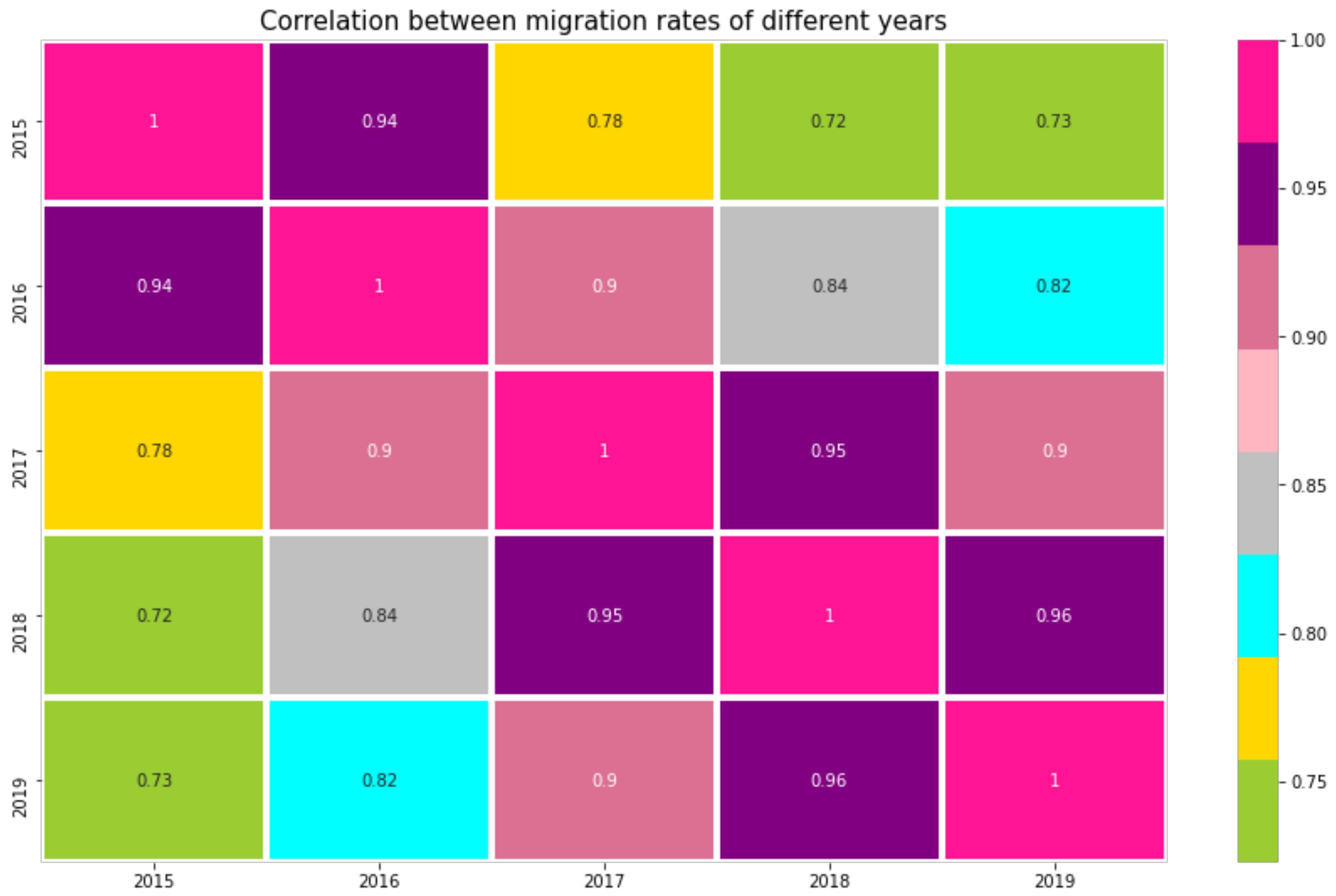
Correlation

In [33]:
```
dt.corr()
```

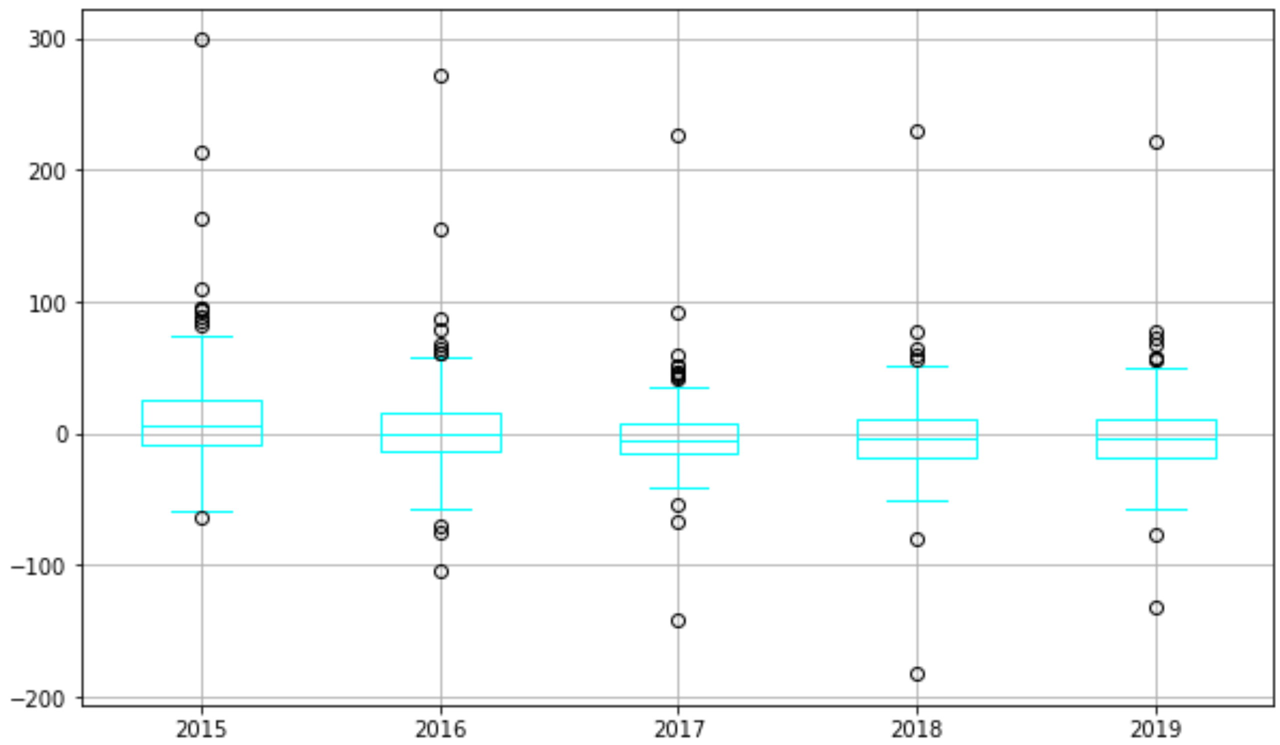
Out[33]:

	2015	2016	2017	2018	2019
2015	1.000000	0.939707	0.783504	0.722801	0.727656
2016	0.939707	1.000000	0.902038	0.837156	0.816151
2017	0.783504	0.902038	1.000000	0.951845	0.900251
2018	0.722801	0.837156	0.951845	1.000000	0.957311
2019	0.727656	0.816151	0.900251	0.957311	1.000000

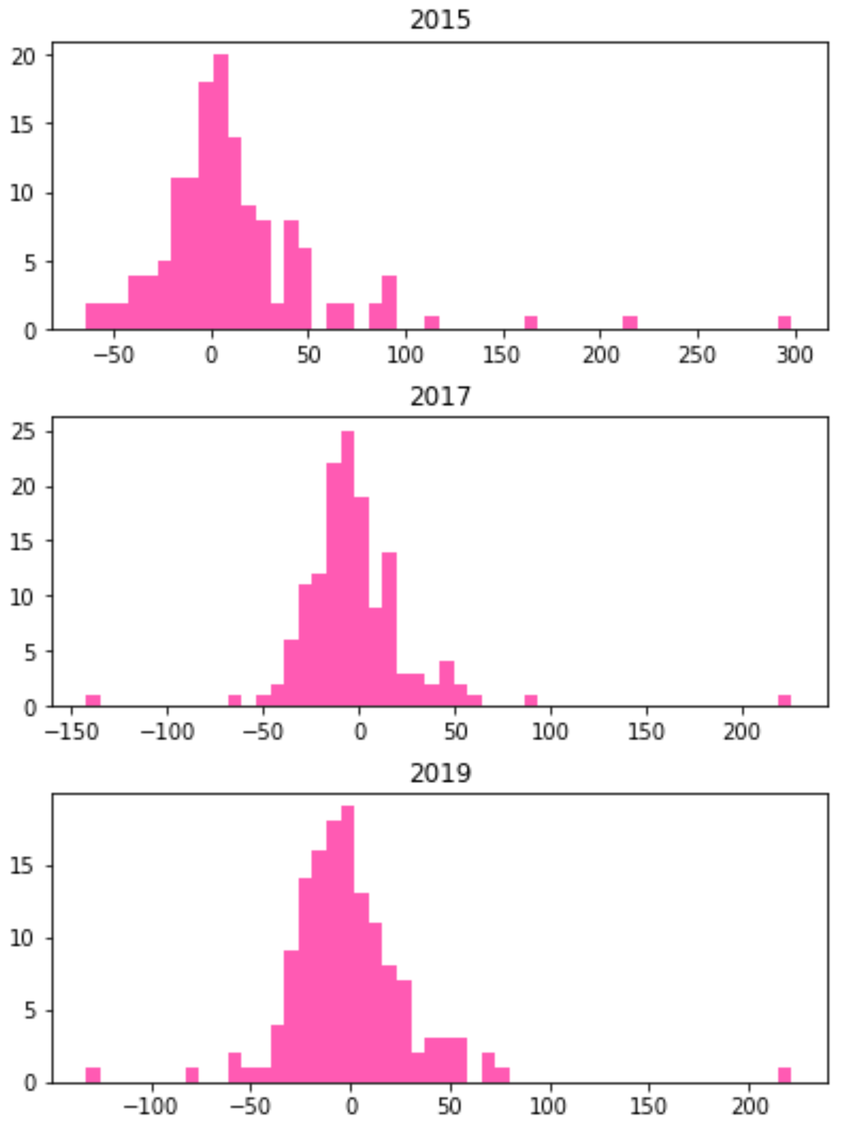
In [34]:
```
plt.figure(figsize=(15,9))
l = ['yellowgreen','gold','cyan', 'silver','lightpink','palevioletred','purple','deeppink']
sns.heatmap(dt.corr(),cmap=l,annot=True,linewidths=3)
plt.title('Correlation between migration rates of different years', fontsize=15)
plt.show()
```



In [35]:
```
plt.figure(figsize=(10,6))
dt.boxplot(color = 'cyan')
plt.show()
```



In [36]:
```
dt.hist(color="deeppink", alpha=0.7, figsize=(15,9), bins=50, grid=False)
plt.show()
```



The last two graphs helped to implement an approximate classification to customise the geographic map in the next section.

---

3. Creating Geographical Map

In [37]:
```
shapefile = '/Users/marine/Downloads/ne_110m_admin_0_countries/ne_110m_admin_0_countries.shp'
```

In [38]:
```
gdf = gpd.read_file(shapefile)[['ADMIN', 'ADM0_A3', 'geometry']]
```

In [39]:
```
gdf.columns = ['country', 'country_code', 'geometry']
gdf.head()
```

Out[39]:

	country	country_code	geometry
0	Fiji	FJI	MULTIPOLYGON (((180.00000 -16.06713, 180.00000...
1	United Republic of Tanzania	TZA	POLYGON ((33.90371 -0.95000, 34.07262 -1.05982...
2	Western Sahara	SAH	POLYGON ((-8.66559 27.65643, -8.66512 27.58948...
3	Canada	CAN	MULTIPOLYGON (((-122.84000 49.00000, -122.9742...
4	United States of America	USA	MULTIPOLYGON (((-122.84000 49.00000, -120.0000...

In [40]:
```
gdf.country
```

Out[40]:
```
0                          Fiji
1      United Republic of Tanzania
2                Western Sahara
3                        Canada
4           United States of America
                  ...
172             Republic of Serbia
173                  Montenegro
174                      Kosovo
175            Trinidad and Tobago
176                 South Sudan
Name: country, Length: 177, dtype: object
```

In [41]:
```
print(gdf[gdf['country'] == 'Antarctica'])
gdf = gdf.drop(gdf.index[159])
```

```
        country country_code  \
159  Antarctica          ATA

                                              geometry
159  MULTIPOLYGON (((-48.66062 -78.04702, -48.15140...
```

In [42]:
```
map_dt = data.groupby('base_country_name').sum().reset_index()
```

In [43]:
```
map_dt.head()
```

Out[43]:

	base_country_name	2015	2016	2017	2018	2019
0	Afghanistan	6.54	5.24	-34.27	15.85	23.01
1	Albania	4.78	0.59	-18.80	-5.05	-16.58
2	Algeria	-5.31	-12.26	-34.10	-23.00	-23.47
3	Angola	73.98	13.90	-9.24	4.62	19.07
4	Argentina	-0.20	2.14	8.19	14.72	-8.77

Corresponding country names from two datasets

In [44]:
```
a = list(map_dt['base_country_name'])
```

In [45]:
```
b = list(gdf['country'])
```

In [47]:
```
not_same_countries = []
for i in b:
    if i not in a:
        not_same_countries.append(i)
```

In [48]:
```
not_same_countries.sort()
```

In [50]:
```
not_same_countries2 = []
for i in a:
    if i not in b:
        not_same_countries2.append(i)
```

In [52]:
```
map_dt['base_country_name'] = map_dt['base_country_name'].replace(['Bahamas, The','Congo, Dem. Rep.','Cz
```

In [54]:
```
map_dt_2019 = map_dt.loc[:, ['base_country_name', '2019']]
```

In [128]:
```
map_dt_2019
```

Out[128]:

	base_country_name	2019
0	Afghanistan	23.01
1	Albania	-16.58
2	Algeria	-23.47
3	Angola	19.07
4	Argentina	-8.77
...	...	...
135	Vietnam	-7.94
136	West Bank and Gaza	-15.59
137	Yemen	2.89
138	Zambia	27.05
139	Zimbabwe	-23.98

140 rows × 2 columns

In [55]:
```
merged = gdf.merge(map_dt_2019, left_on = 'country', right_on = 'base_country_name', how = 'left')
```

In [56]:
```
merged.tail()
```

Out[56]:

	country	country_code	geometry	base_country_name	2019
171	Republic of Serbia	SRB	POLYGON ((18.82982 45.90887, 18.82984 45.90888...	NaN	NaN
172	Montenegro	MNE	POLYGON ((20.07070 42.58863, 19.80161 42.50009...	NaN	NaN
173	Kosovo	KOS	POLYGON ((20.59025 41.85541, 20.52295 42.21787...	NaN	NaN
174	Trinidad and Tobago	TTO	POLYGON ((-61.68000 10.76000, -61.10500 10.890...	Trinidad and Tobago	-13.81
175	South Sudan	SDS	POLYGON ((30.83385 3.50917, 29.95350 4.17370, ...	NaN	NaN

In [57]:
```
merged.fillna('No data', inplace = True)
```

In [58]:
```
import json
merged_json = json.loads(merged.to_json())
json_data = json.dumps(merged_json)
```

In [60]:
```
from bokeh.io import output_notebook, show, output_file
from bokeh.plotting import figure
from bokeh.models import GeoJSONDataSource, LinearColorMapper, ColorBar
from bokeh.palettes import brewer
```

In [61]:
```
geosource = GeoJSONDataSource(geojson = json_data)
```

In [62]:
```
palette = brewer['YlGnBu'][6]
```

In [63]:
```
palette = palette[::-1]
```

In [64]:
```
color_mapper = LinearColorMapper(palette = palette, low = -60, high = 60, nan_color = '#d9d9d9')
```

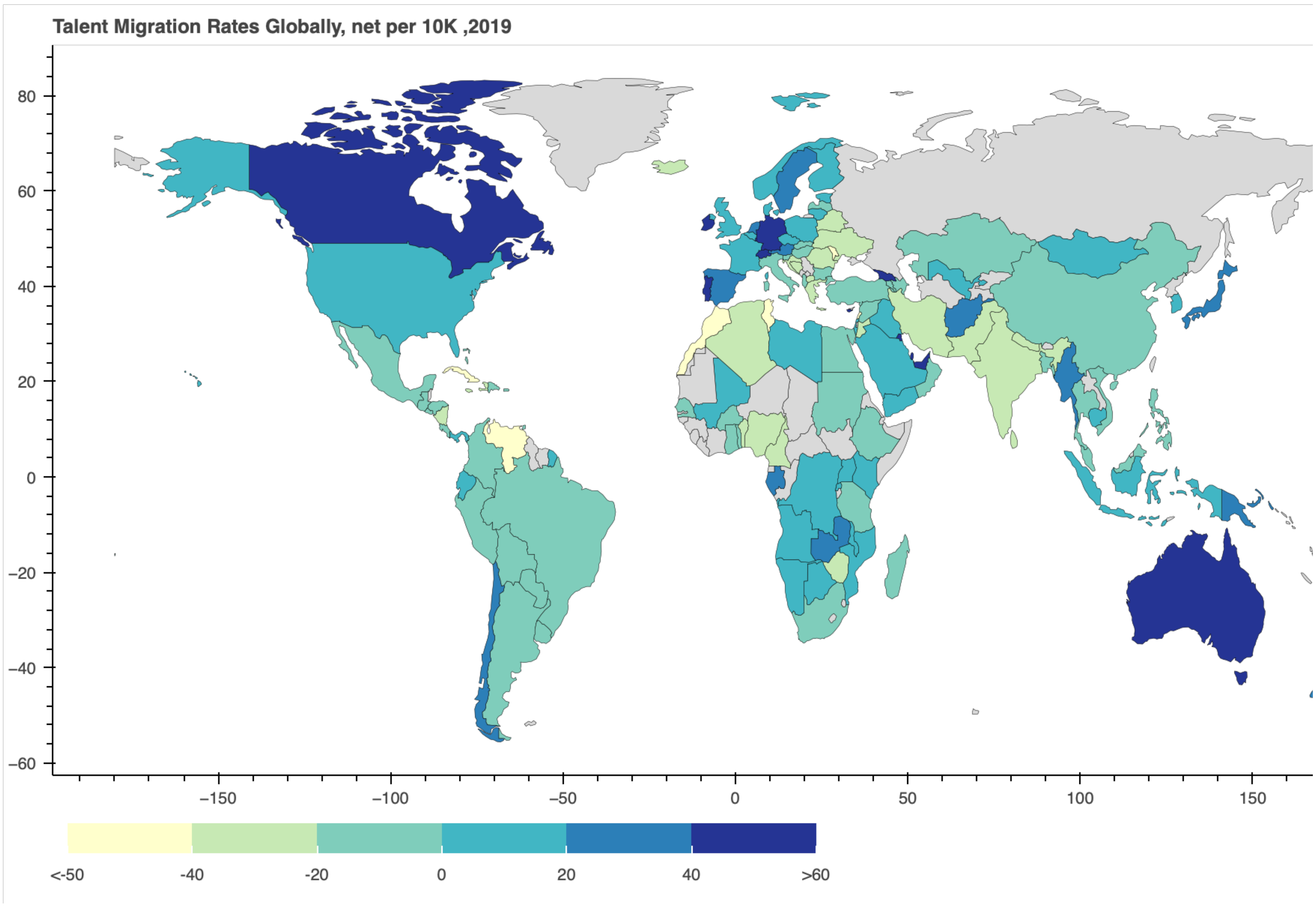
In [65]:
```
tick_labels = {'-60': '<-50', '-40': '-40', '-20':'-20', '0':'0', '20':'20', '40':'40', '60':'>60',
               '300':'300', '50': '>40'}
```

In [66]:
```
color_bar = ColorBar(color_mapper=color_mapper, label_standoff=8,width = 500, height = 20,
border_line_color=None,location = (0,0), orientation = 'horizontal', major_label_overrides = tick_labels
```

In [67]:
```
p = figure(title = 'Talent Migration Rates Globally, net per 10K ,2019', plot_height = 600 , plot_width
           toolbar_location = None)
p.xgrid.grid_line_color = None
p.ygrid.grid_line_color = None
```

In [68]:
```
p.patches('xs','ys', source = geosource,fill_color = {'field' :'2019', 'transform' : color_mapper},
          line_color = 'black', line_width = 0.25, fill_alpha = 1)
p.add_layout(color_bar, 'below')
output_notebook()
show(p)
```

(https://bokeh.org) BokehJS 2.3.0 successfully loaded.



Missing data on the map is filled with light grey.

4. Extracting Insights from Armenian Talent Migration

In [69]:
```
Armenia = data.loc[data['base_country_name'] == 'Armenia']
```

In [70]:
```
Armenia
```

Out[70]:

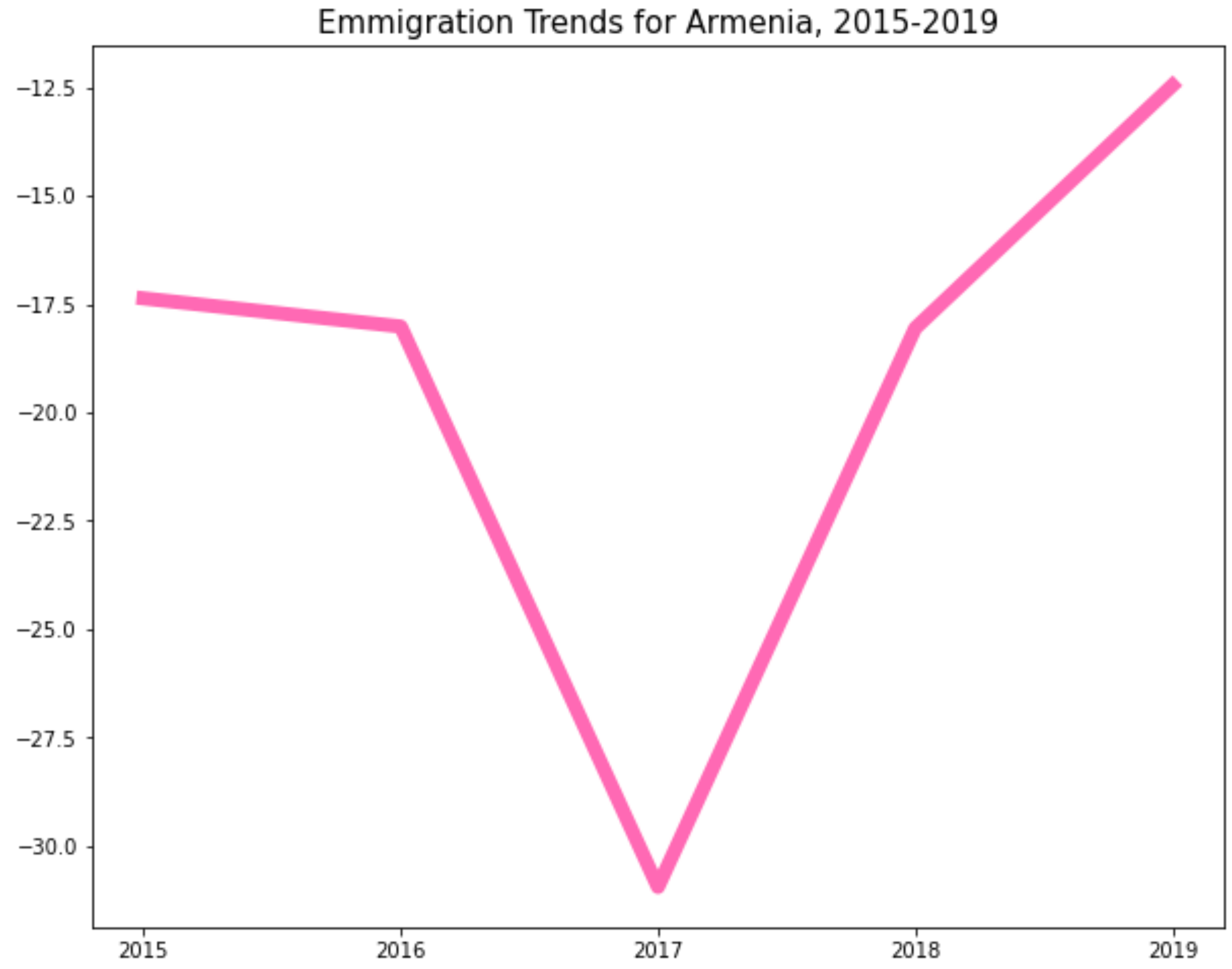
	base_country_code	base_country_name	base_country_wb_income	base_country_wb_region	target_country_code	target_country_name	ta
128	am	Armenia	Upper Middle Income	Europe & Central Asia	ca	Canada	
129	am	Armenia	Upper Middle Income	Europe & Central Asia	fr	France	
130	am	Armenia	Upper Middle Income	Europe & Central Asia	de	Germany	
131	am	Armenia	Upper Middle Income	Europe & Central Asia	ir	Iran, Islamic Rep.	
132	am	Armenia	Upper Middle Income	Europe & Central Asia	ae	United Arab Emirates	
133	am	Armenia	Upper Middle Income	Europe & Central Asia	gb	United Kingdom	
134	am	Armenia	Upper Middle Income	Europe & Central Asia	us	United States	

In [71]:
```
overall_rates = dict(Armenia.sum()[8:13])
```

In [72]:
```
plt.figure(figsize=(10,8))
plt.plot(overall_rates.keys(), overall_rates.values(), c= 'hotpink', lw = 7)


plt.title('Emmigration Trends for Armenia, 2015-2019', fontsize=15)

plt.show()
```



In [73]:
```
Armenia['Average Rate'] = Armenia.loc[:,'2017':'2019'].mean(axis=1)
```
...

In [75]:
```
sum_ = Armenia['Average Rate'].abs().sum()
```

In [76]:
```
percentage_ = [abs(i)/sum_*100 for i in Armenia['Average Rate']]
```

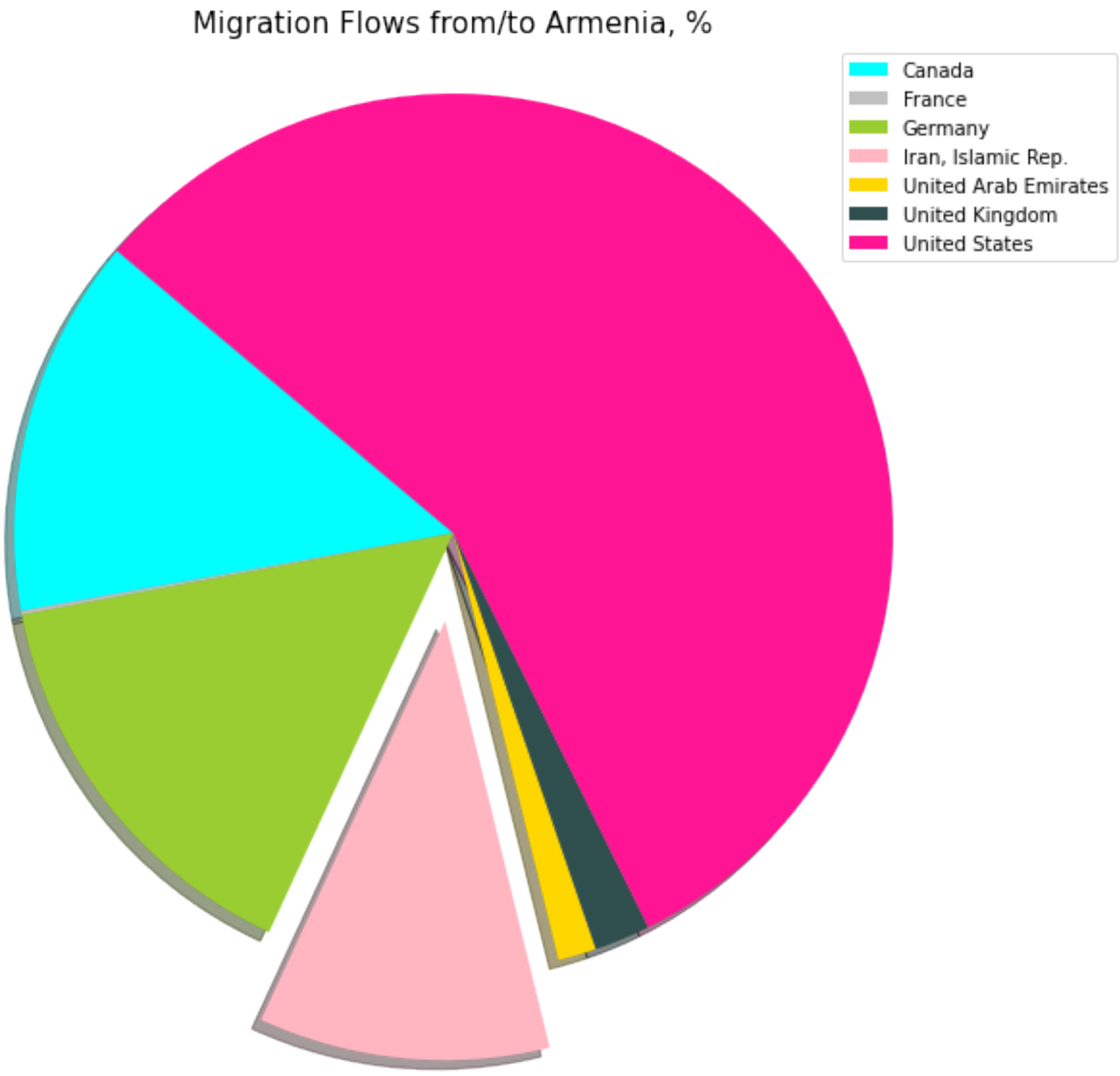
In [78]:
```
Armenia['Percentage'] = percentage_
```
...

In [79]:
```
list(Armenia['target_country_name'])
```

Out[79]:
```
['Canada',
 'France',
 'Germany',
 'Iran, Islamic Rep.',
 'United Arab Emirates',
 'United Kingdom',
 'United States']
```

In [80]:
```
labels = list(Armenia['target_country_name'])
rates = list(Armenia['Percentage'])
colors = ['cyan', 'silver','yellowgreen','lightpink', 'gold', 'darkslategrey','deeppink','palevioletred'
explode = (0, 0, 0, 0.2, 0, 0, 0)

plt.figure(figsize=(10,8))
patches, texts = plt.pie(rates, explode=explode, colors=colors, shadow=True, startangle=140)
plt.legend(patches, labels, loc="best")
plt.axis('equal')
plt.tight_layout()
plt.title('Migration Flows from/to Armenia, %', fontsize=15)
plt.show()
```



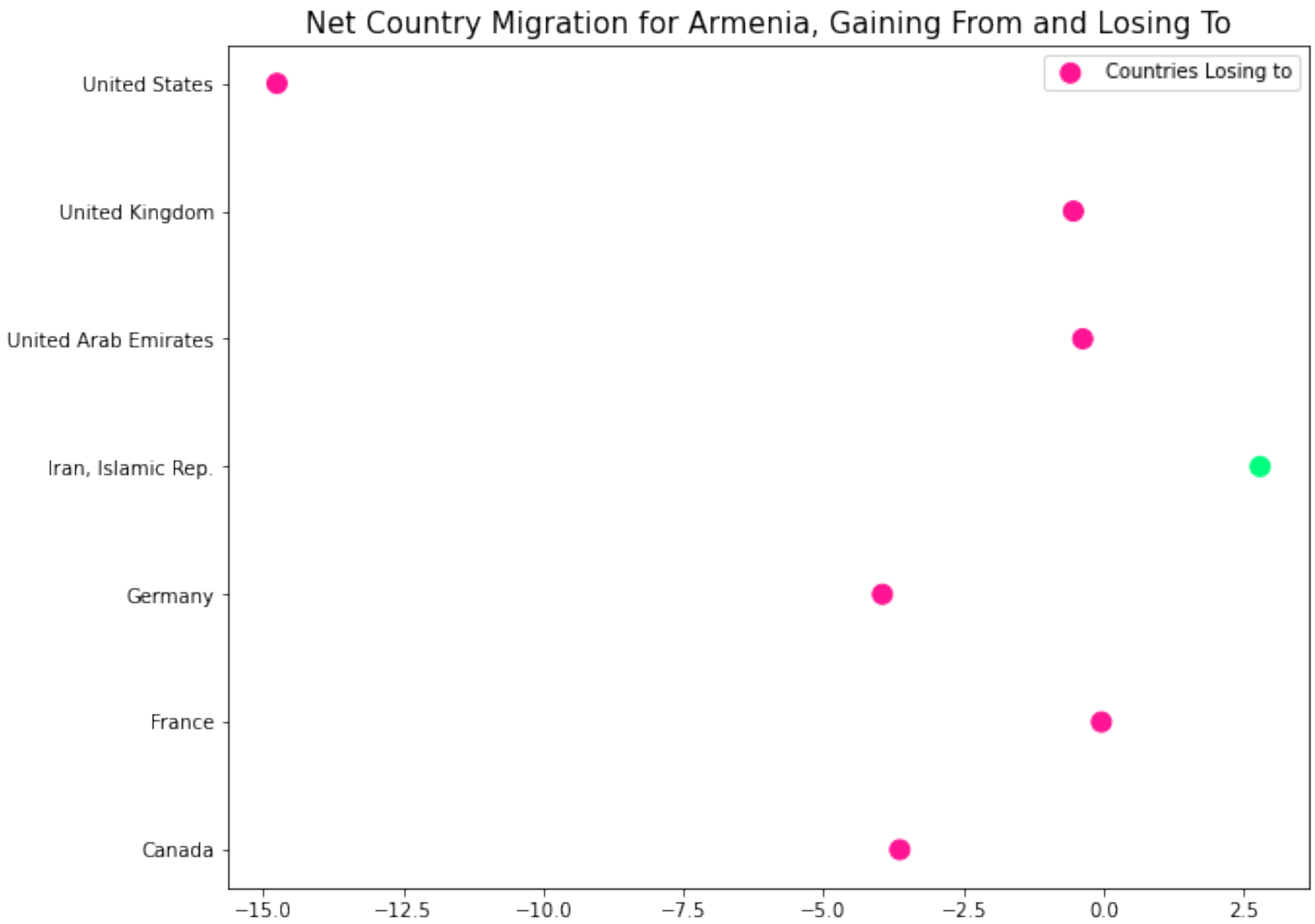
Exploded part represents destinations for which net migration rate is positive(flows to Armenia are greater).

In [81]:
```
import matplotlib.colors as mcolors

colors = np.where(Armenia["Average Rate"]>0,'springgreen','deeppink')
classes = 'Countries Losing to'
classes2 = 'Countries Gaining From'

plt.figure(figsize=(10,8))
plt.scatter(list(Armenia['Average Rate']), list(Armenia['target_country_name']), c = colors, s=100, labe

plt.legend(fontsize=10)
plt.title('Net Country Migration for Armenia, Gaining From and Losing To', fontsize=15)
plt.show()
```



Please note that the dataset does not include net migration rates of Russia.

```
Explanation: Per 10,000 citizens, about 15 more people emmigrated from Armenia to US than immigr
ated to Armenia during 2017-2019. For the same period, approximately 3 more people entered Armen
ia from Iran than left.
```

---

5. Exploring Flow Directions (based on income groups)

Usually migration flows occur from low, lower-middle, upper-middle income countries to high income ones. Let's check it out.

In [82]:
```
high_income = data.loc[data['base_country_wb_income'] == 'High Income']
```

In [83]:
```
income = high_income.target_country_wb_income.unique()
```

In [85]:
```
high_income
```

Out[85]:

	base_country_code	base_country_name	base_country_wb_income	base_country_wb_region	target_country_code	target_country_name
0	ae	United Arab Emirates	High Income	Middle East & North Africa	af	Afghanistan
1	ae	United Arab Emirates	High Income	Middle East & North Africa	dz	Algeria
2	ae	United Arab Emirates	High Income	Middle East & North Africa	ao	Angola
3	ae	United Arab Emirates	High Income	Middle East & North Africa	ar	Argentina
4	ae	United Arab Emirates	High Income	Middle East & North Africa	am	Armenia
...	...	...	...	...	...	...
3994	uy	Uruguay	High Income	Latin America & Caribbean	pe	Peru
3995	uy	Uruguay	High Income	Latin America & Caribbean	es	Spain
3996	uy	Uruguay	High Income	Latin America & Caribbean	gb	United Kingdom
3997	uy	Uruguay	High Income	Latin America & Caribbean	us	United States
3998	uy	Uruguay	High Income	Latin America & Caribbean	ve	Venezuela, RB

2415 rows × 13 columns

In [87]:
```
# Overall flows
```

In [88]:
```
d = {}
for i in income:
    inc = i.split()[:-1]
    a = high_income.loc[(high_income['target_country_wb_income']==i)]
    if len(inc) == 2:
        b = str(inc[0]).lower()+'_'+str(inc[1]).lower()+'_to_high'
    else:
        b = str(inc[0]).lower()+'_to_high'
    d[b] = len(a)
    a = 0
    b = 0
d
```

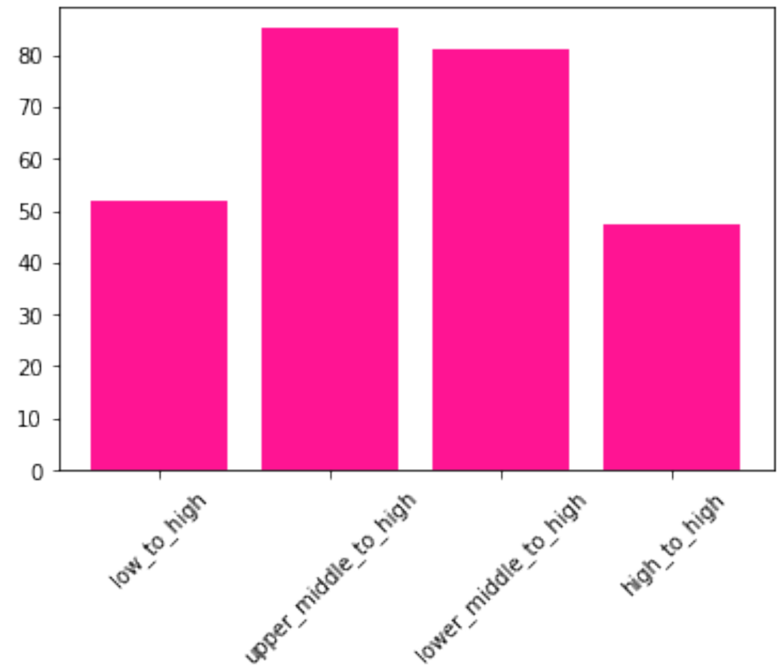
Out[88]:
```
{'low_to_high': 104,
 'upper_middle_to_high': 552,
 'lower_middle_to_high': 387,
 'high_to_high': 1372}
```

In [89]:
```
def in_out_flows(year):
    d_ = {}
    for i in income:
        inc = i.split()[:-1]
        a = high_income.loc[(high_income['target_country_wb_income']==i)
            & (high_income[str(year)] > 0)]
        if len(inc) == 2:
            b = str(inc[0]).lower()+'_'+str(inc[1]).lower()+'_to_high'
        else:
            b = str(inc[0]).lower()+'_to_high'
        prc = (len(a)/d[b])*100
        d_[b] = round(prc, 2)
        a = 0
        b = 0
    return d_
```

In [91]:
```
for i in range(2015, 2020):
    print(i)
    print(in_out_flows(i))
```

```
2015
{'low_to_high': 39.42, 'upper_middle_to_high': 66.67, 'lower_middle_to_high': 61.24, 'high_to_high': 48
.18}
2016
{'low_to_high': 45.19, 'upper_middle_to_high': 78.26, 'lower_middle_to_high': 70.8, 'high_to_high': 48.
83}
2017
{'low_to_high': 55.77, 'upper_middle_to_high': 83.51, 'lower_middle_to_high': 79.07, 'high_to_high': 48
.25}
2018
{'low_to_high': 56.73, 'upper_middle_to_high': 81.88, 'lower_middle_to_high': 78.04, 'high_to_high': 48
.83}
2019
{'low_to_high': 51.92, 'upper_middle_to_high': 85.14, 'lower_middle_to_high': 80.88, 'high_to_high': 47
.52}
```

In [92]:
```
plt.bar(*zip(*in_out_flows(2019).items()), color = 'deeppink')
plt.xticks(rotation = 45)
plt.show()
```



Interestingly, according to the data, the number of inflows to low income from high income countries was even higher than the other way round in both 2015 and 2016, and then it almost equals between 2017 and 2019.